软件测试—集成测试

集成测试的定义

集成测试就是在单元测试的基础上，将所有已通过单元测试的模块按照概要设计的要求进行组装，组装为子系统或者系统，执行测试，目的是确保各单元模块组合在一起后能够按既定意图协作运行

集成测试的内容

1. 将各个具有相互调用关系的模块组装起来时，检查穿越模块接口的数据是否会丢失
2. 判断各子功能组合起来能否达到预期要求的父功能
3. 检查一个模块的功能是否会对其他模块的功能产生不利影响
4. 检查全局数据结构是否正确，以及在完成模块功能的过程中是否会被异常修改
5. 单个模块的误差积累起来，是否会放大到不可接受的程度

集成测试的评价

1. 测试用例的规模，越少越好
2. 桩模块的设计，越少越好
3. 驱动模块的设计，越少越好
4. 缺陷的定位，集成测试用到的接口数量越少越好

几种集成方法

1.成对集成

选择与测试单元有依存关系的模块一同测试，但每次只涉及一对接口，较容易定位缺陷，成对集成最初的目的是避免开发，驱动模块与桩模块，但是到真实的测试场景中，还是无法避免开发对应的模块

2.邻居集成

邻居集成的基本思想是将每个集成测试用例限定在某个节点的邻居上，针对某个模块的集成测试用例应包含该模块与其邻居。所谓邻居，就是模块的一个特定邻域模块集合，即包括调用它的上层模块，也包括它所调用的下级模块

邻居集成试图通过扩大单个测试用例的范围来减少测试用例的总数，导致缺陷定位变得困难

3.独立路径的集成

将函数的调用图看为程序的控制流图或者程序图，从根节点到子节点的调用形成了路径，每条独立路径看为一个集成测试用例

集成测试遍历顺序的设计

1.大爆炸集成

将所有经过单元测试的模块一次性拼装到被测系统中进行测试，完全不考虑模块间的依赖性和可能存在的风险

2.自顶向下集成

是从主控模块开始，按照系统的顺序结构，沿着控制层次自上而下，逐渐拼装起来。该方法无需开发驱动模块，但是对未集成的的模块，**需要开发桩模块。**采用广度优先或者深度优先的方式向下推进。

步骤

1. 对根节点进行集成测试，所有被根节点直接调用的模块均用桩模块来代替
2. 根据所选择的推进策略，用实际模块逐渐替换桩模块（理论上一次替代一个）。并用新的桩模块代替新加入的模块，与已测模块或子系统构成新的子系统，进行测试
3. 回归测试，全部或者部分执行以前做过的测试，以确保新加入模块中未引入新的缺陷
4. 重复 2 3 直到所有模块都已集成到系统中

3.自底而上的集成

从底层模块（叶子节点）开始，按照图的调用结构，从下而上，逐渐将各模块组装起来。该方法无需开发桩模块，需要针对未经集成测试的模块**开发驱动模块**，以广度优先或者深度优先推进

步骤

与自顶向下类似，不再赘述

4.三明治集成

此策略是自顶向下与自底而上的策略都使用的一种测试策略，在子树上再进行大爆炸集成

三种策略

1. 选取目标层，目标层之上的使用自顶而下的策略，目标层之下的使用自底而上的策略，具有两种测试方式的优点，可降低桩模块与驱动模块的开发，但最终会因为在目标层上使用大爆炸集成，导致目标层测试不充分
2. 基于策略1，并对目标层使用独立测试的策略，缺点是引入了更多的测试用例，提高了桩模块与驱动模块的开发量
3. 对**包含读操作**的子系统**自底而上**直到根节点，对**包含写操作**的子系统**自顶而下**直至叶子节点，实质是将输入输出于被测系统结合，也可看为是输入输出优先策略

比较

项目	测试用例数目	桩模块	驱动模块	缺陷定位	并行测试	系统概貌
成对集成	由边数决定	需要	需要	非常容易	可以	不确定
邻居集成	主要由中间的节点决定	需要	需要	困难	可以	不确定
大爆炸集成	少	不需要	不需要	非常困难	N/A	早期
自顶而下	较多	需要	不需要	较容易	困难	早期
自底而上	较多	不需要	需要	较容易	可以	较晚
三明治	较多	需要	需要	较困难	可以	早期